JVM 进行垃圾回收的过程主要有这么几个：

1. 判断对象是否可达
2. 如果不可达，进行第一次标记
3. 第二次检测对象还是不可达，进行第二次标记
4. 如果对象被两次标记，说明对象已死，对象将被回收

* 如何判断对象已死？

JVM使用可达性分析进行对象已死的判断。

* 可达性分析（Reach-ability Analysis）：从GC Root 开始搜索，所有走过的路径叫做引用链(Reference Chain). 当对象不在任何一个引用链中时，对象将被标记为不可达。
* GC Root 包括以下几种：
* 虚拟机栈中引用的对象
* 方法区中类静态属性引用的对象
* 方法区中常量所引用的对象
* 本地方法栈中的JNI(即一般说的native 方法)引用的对象
* 对象的引用按照垃圾回收的优先级有低到高分别是：
* 强引用(Strong Reference)：只要存在强引用就不会被回收
* 软引用(Soft Reference)：在将要发生内存溢出异常之前会被回收
* 弱引用(Weak Reference)：发生垃圾回收就会被回收
* 虚引用(Virtual Reference)：唯一的目的是在这个对象被回收时，会收到一个系统通知
* 在第二次标记之前，如果将对象变成可达，对象将从回收列表中删除。对象的finalize方法时对象逃脱回收的最后一次机会。还有就是任何一个对象的finalize方法只会执行一次。
* 对于方法区的回收，主要回收的对象是废弃常量和无用的类。
* 废弃常量：在当前线程中没有任何一个一样值的变量。
* 无用的类：不存在该类的实例，类加载器被回收，无法再任何地方通过方式方法类的方法。
* 垃圾回收的算法
* 标记-清除(Mark - Sweep)：对已死对象进行标记，根据标记将对象进行回收。缺点：如果内存被频繁的分配或者是回收，将会有很多内存碎片。
* 复制(Copying)：将内存进行划分多部分，预留一部分空闲内存。在内存回收时，将存活的对象拷贝到预留的那部分空闲内存中。并回收原来那部分的所有内存。
* 标记-整理(Mark - Compact)：将存活的的对象都向内存的一端移动，然后直接清理端边界以外的所有内存。